Easy Ropes 2D

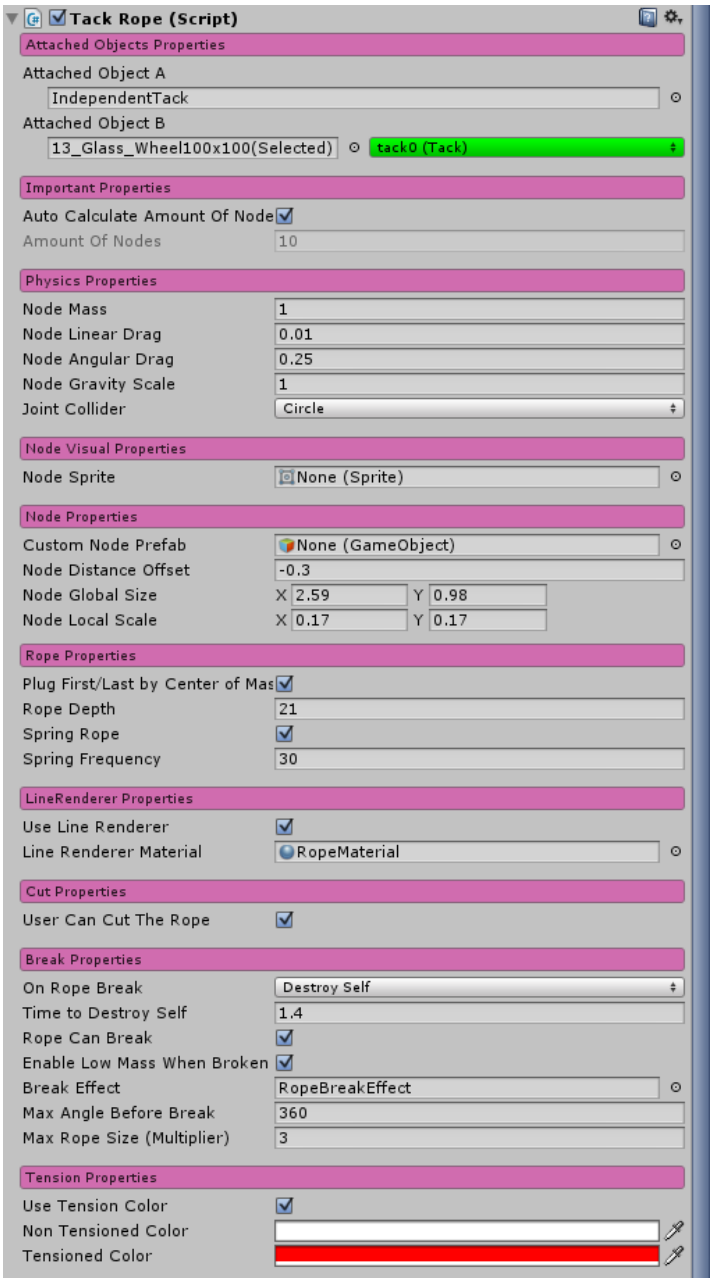
Powered by KILTASSETS

Version 1.1

raf.csoares@gmail.com

1. TACK ROPE (COMPONENT)

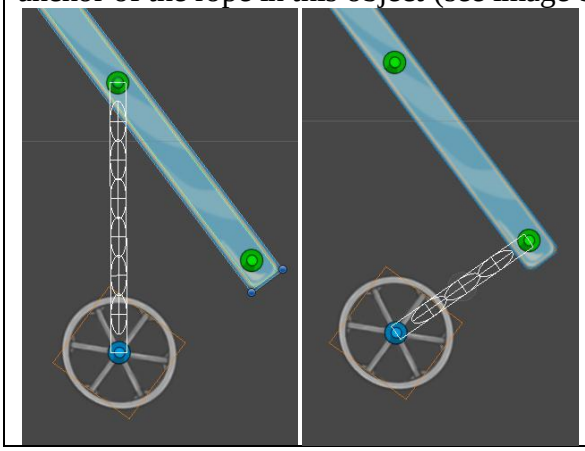
This is the main rope component in this package. Use this ***MonoBehaviour*** to easily instantiate a ***Rope2D*** in your scene.



PROPERTIES

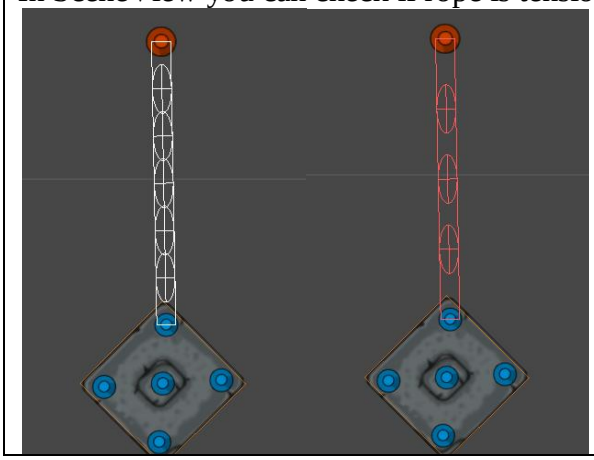
Attached Object A/Attached Object B:

GameObject field that represents First and Last Points to rope use as reference. These objects must have a ***Rigidbody2D*** inside their hierarchy. If the hierarchy of the object contains a ***TacksInventory*** component a combo box will be displayed. This combo box can be used to change anchor of the rope in this object (see image below).



Auto Calculate Amount of Nodes/Amount of Nodes:

Use these properties to control amount of nodes in rope. Uncheck ***Auto Calculate Amount of Nodes*** to generate Tension in rope.
In SceneView you can check if rope is tensioned (see image below):



Node Mass/Node Linear Drag/Node Angular Drag/Node Gravity Scale:

Rigidbody2D properties of each node in rope. Change this to manipulate intensity of the force applied by the rope and to force rope to lose energy over time.
The tenacity of rope is proportional to ***Node Mass*** so you can't create a rope with no mass.

- **Joint Collider:**

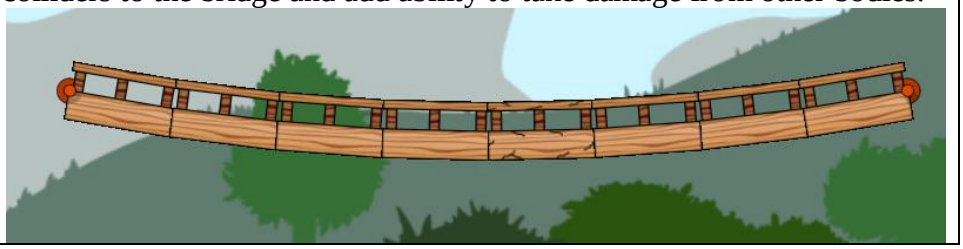
Checkbox to select the collider type of each node(Box or Circle).
If you need custom colliders in your node you must use *Custom Node Prefab* (see below)

- **Node Sprite:**

Use this to Select which ***Sprite*** will be used in each node when created.
This sprite will only be used if *Use Line Renderer* mode is turned off

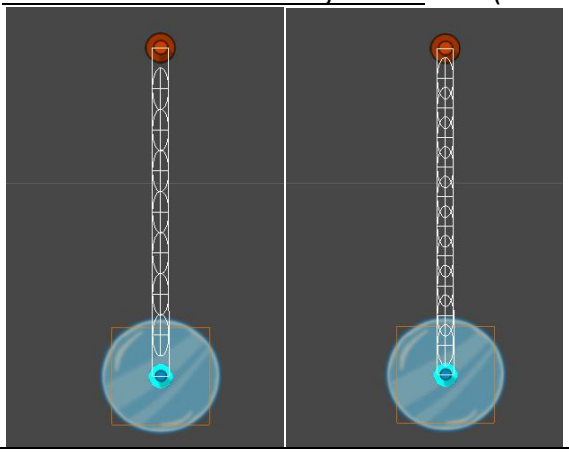
- **Custom Node Prefab:**

Passing a ***GameObject*** as a model, you will be able to customize how nodes in this rope will act.
In ***02_BridgeScene*** we used a custom node to implement custom colliders to the bridge and add ability to take damage from other bodies.



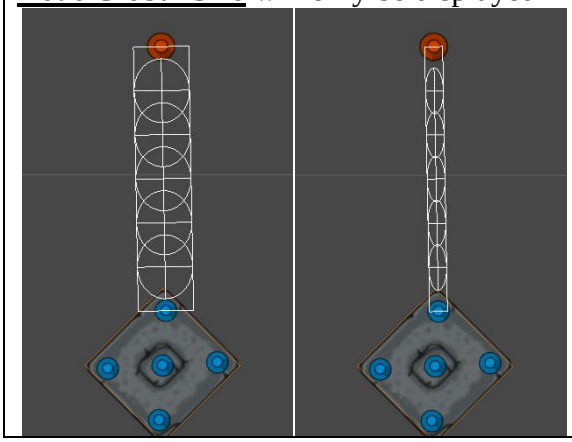
- **Node Distance Offset:**

Distance between nodes. This will affect the amount of nodes created if *Auto Calculate Amount of Nodes* is on(see Image below).



- **Node Local Scale/ Node Global Size:**

Use this properties to control the size of each node in rope.
Node Global Size will only be displayed if ***Node Sprite*** is empty.

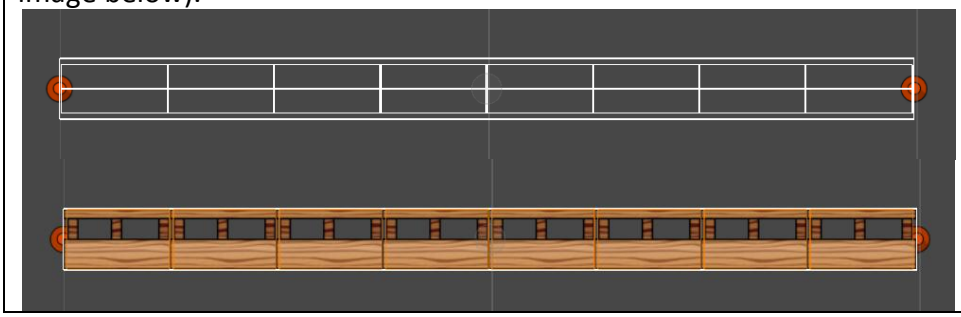


- **Plug First/Last by Center of Mass:**

If enabled, the first and the last node of the rope will be plugged by the center of mass(see image below). Ropes with this property enabled will be more stable then ropes with this property disabled.
Note: only disable this property when not using ***LineRenderer*** or if node ***Collider2D*** is too big):

- **Editor Visual Option:**

Check Box with two options ('Gizmos' or 'NodeObjects'). If 'Gizmos' is enabled, the rope will only be created when game starts. If 'NodeObjects' is enabled the nodes will be created in SceneView(see image below).



- **Rope Depth:**

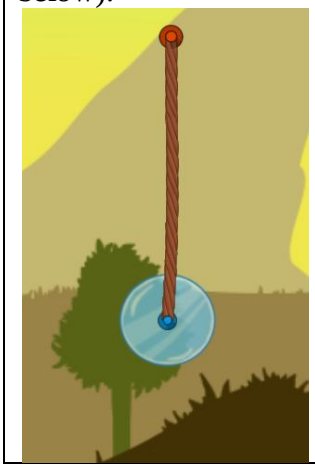
Control the sorting order of each node/***LineRenderer*** of this rope. Useful if you need to display rope above/below any ***Sprite*** in scene.

- **Spring Rope/ Spring Frequency:**

If this property is enable the nodes will be created with ***SpringJoint2D*** instead of ***DistanceJoint2.*** Spring Ropes can be used to simulate sling effects(See ***01_SlingScene***)

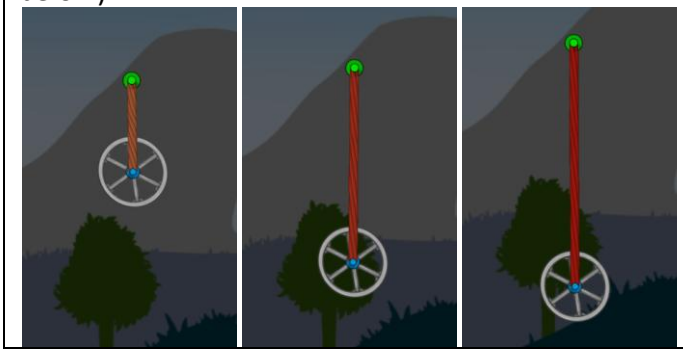
- **Use Line Renderer/Line Renderer Material:**

With ***LineRenderer*** the rope can be displayed by a mesh(see image below).



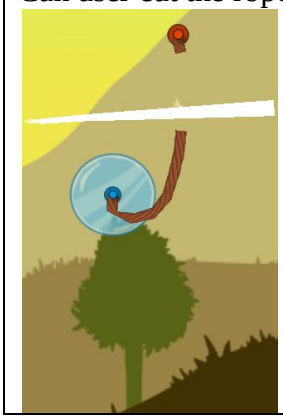
- **Use Tension Color/ Non-Tensioned Color/ Tensioned Color:**

If this property is enable, the rope will change its color based on tension. When rope size <= normal rope size the color will be ***Non-Tensioned Color***. When rope size >= (**Max Rope Size** * normal rope size) the color will be ***Tensioned Color***. Values between this will display a color based in how much the rope has been tensioned(see image below).



- **User Can Cut The Rope:**

Can user cut the rope with Cutter?



- **Rope Can Break:**

Enable this to use ***Max Rope Size*** and ***Max Angle Before Break***

- **Max Angle Before Break:**

Max possible angle of one node (based in your next and previous node) before this node break.

- **Max Rope Size (Multiplier):**

Max rope size before break. ex: 3 = three times normal rope size.

- **On Rope Break/Time to Destroy Self:**

Combo Box with two options: 'Destroy Self' or 'Nothing'.
If 'Destroy Self' is selected, all nodes will be destroyed, using a fade effect, after ***Time to Destroy Self*** seconds when user cut the rope or any node break.

- **Enable Low Mass When Broken:**

This special mode can be used to try to simulate ropes with no mass. When rope breaks, the mass of each node will be changed to zero.

- **Break Effect:**

Prefab instantiated by rope when a node breaks (Don't forget to destroy this effect after some time or it will be inside your scene forever. If this effect is a particle you can add ***AutoDestroyParticle*** component in root).

CALLBACKS

- **OnRopeDestroyed (*Rope2D* p_ropeDestroyed)**

  Called when rope is destroyed (Before OnDestroy).

- **OnRopeCreated (*Rope2D* p_ropeCreated)**

  Called when the rope is created by function 'CreateRope'.

- **OnRopeBreak (*Rope2D* p_rope, *int* p_nodeIndex)**

  Called when any node has been destroyed.

- **OnRopeCut (*Rope2D* p_rope, *int* p_nodeIndex)**

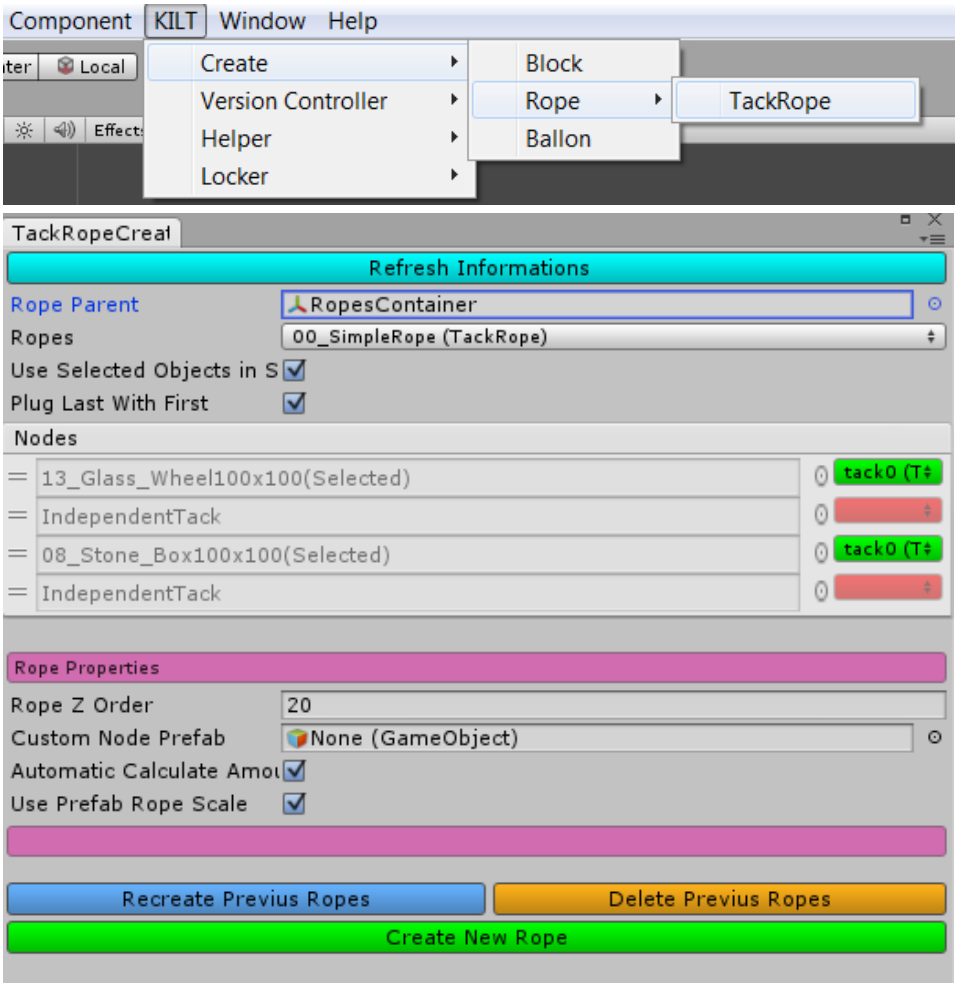
  Called when the user cut the rope. ***OnRopeBreak*** will be called before ***OnRopeCut***.

- **OnRopeTensioned (*Rope2D* p_rope, *float* p_deltaTension)**

  Called when rope is tensioned. 'p_*deltaTension*' is a value between 0 and 1. If 'p_*deltaTension*' equals one and ***RopeCanBreak*** is enabled, the rope will break.

2. TACK ROPE CREATOR (EDITOR WINDOW)

This window can be used to create new ropes in game. Just select, in scene, which objects you want to plug by a rope, select rope prefab and click in 'Create New Rope' button.



▪ PARAMETERS

➢ Rope Parent

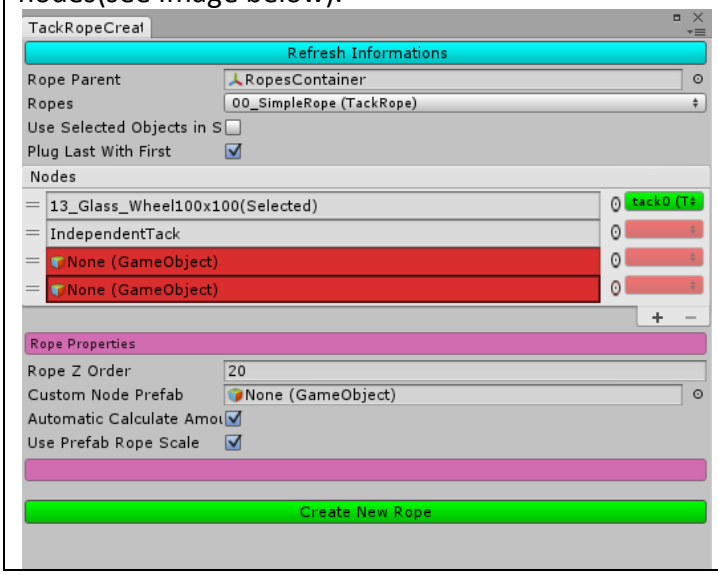
This will be the parent of ropes created in this scene. The scene will try to find a ***GameObject*** with name "RopesContainer", if not found you must pass this parameter by yourself, or keep the parent equals null.

➢ Ropes

ComboBox with all ***TackRope*** Prefabs in project. This will be the prefab instantiated when creating rope.

- **Nodes / Use Selected Objects in Scene / Plug Last with First**

Nodes are all objects to plug ropes. You can reorder then to create ropes in correct order. If ***Use Selected Objects in Scene*** is enabled, the objects in this list will be the objects selected in scene. You can plug last with first points by toggling the checkbox.
If ***Use Selected Objects in Scene*** is disabled you must drop your own nodes(see Image below).



- **Rope Z Order / Custom Node Prefab / Automatic Calculate**

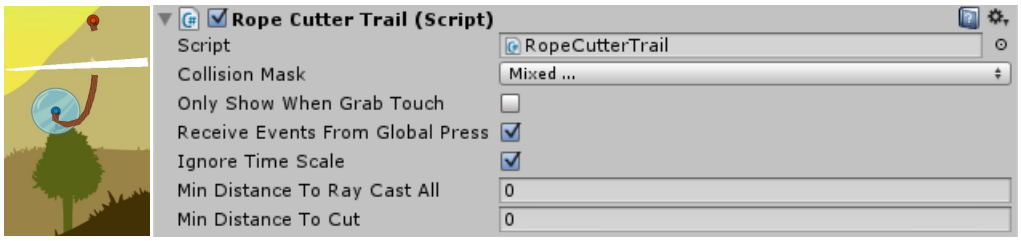
These properties will override the default prefab properties with same name (Note: ***Rope Z Order*** equals ***Rope Depth***)

- **Use Prefab Rope Scale**

Take care with this property. If your prefab was created inside an parent and you want to keep local scale saved you must keep this option enabled.
Note: All prefabs included in this asset need this option enabled.

3. ROPE CUTTER (COMPONENT)

This component can be used to cut ropes in scene. A prefab can be found in this asset so you can use it or implement your own cutter based on this cutter.



- **DEPENDENCES**

  RopeCutter receive events from ***KiltUICamera*** so you must have one of this Components attached to your Camera. If you have NGUI Camera in your scene you can replace ***UICamera*** for ***KiltUICamera*** (***KiltUICamera*** will inherit from ***UICamera*** if NGUI is detected).

  Note: Don't worry about new UGUI system. ***KiltUICamera*** will work with new UGUI too.

- **PROPERTIES**

  - **Collision Mask**

    Combo Box with all possible layers in this project. Only selected layers will collide with Cutter's RayCast

  - **Only Show When Grab Touch**

    If this option is enabled, the cutter will only be activate if user click don't hit other objects (returned by ***KiltUICamera)***.

  - **Receive Events From Global Press**

    If this option is enabled, the cutter will register events from ***Global Press Controller*** instead of ***KiltUICamera*** (only if ***Global Press Controller*** instance exists in scene).

  - **Ignore Time Scale**

    Will cutter works even when paused?

  - **Min Distance to Ray Cast All**

    Min distance, in global scale, to call function to Ray Cast All. Remember that cutter can still work with Unity's collision detection event, so you don't need RayCastAll enabled all time if you have a collider attached to cutter.

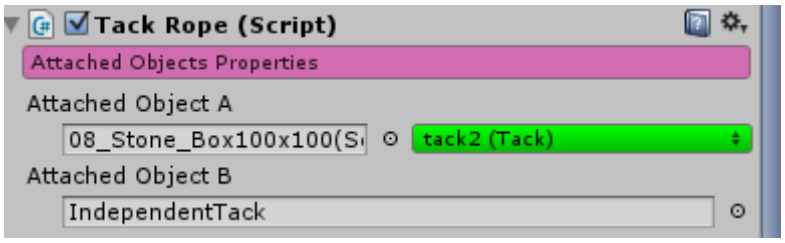
  - **Min Distance To Cut**

    Min Distance, in global scale, to enable cutter.

4. HOW TO CREATE PLUGGABLE OBJECTS

Pluggable Objects must have a ***Rigidbody2D*** attached in any ***GameObject*** inside it hierarchy. This object MUST have ***Collider2D*** with ***Trigger*** disabled. This is enough to plug one rope to this object, but if you want to make possible change plugged anchor you must create an object with component ***TacksInventory*** and place objects inside this inventory with component ***Tack*** attached.

When an object without tack was selected to be one attached point in rope, no combo box will be displayed (see image below).



Note: Inside asset folder ***Examples->Extra->Blocks->Resources->Prefabs->_TacksInventoryModel*** you can find simple models of ***TacksInventory.*** Just drop this model inside your object hierarchy and change position/create new tacks to it.

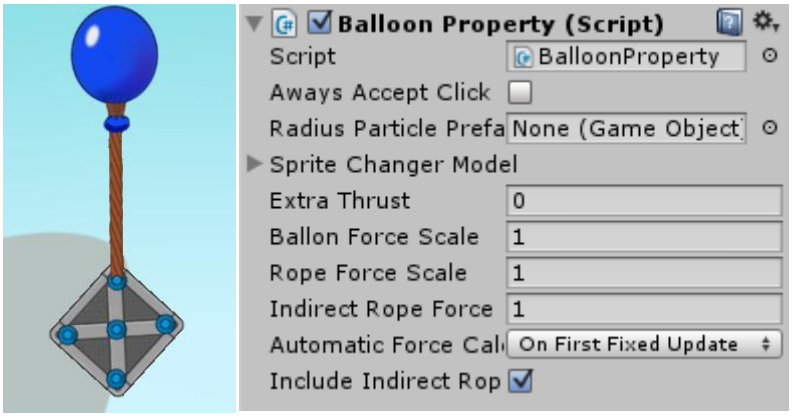
5. PLUGGING BALLOONS TO OBJECTS

This package comes with an extra component called ***BalloonProperty***, this component can be used to calculate an amount of force to support objects attached by rope (see images below).

- **COMPONENT**

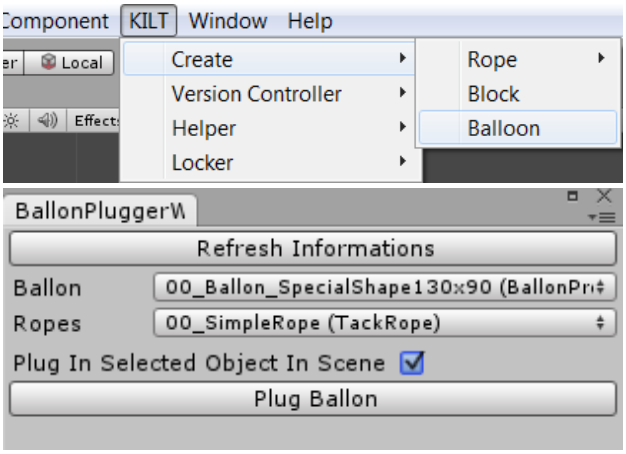
  By default a balloon calculate the exactly amount of force needed to support one object in air (negating rope mass and simulating zero mass rope).

  You can change force scales to manipulate balloon automatic forces, or add ***Extra Thrust***(in newtons).



- **EDITOR WINDOW**

  Similar to ***Tack Rope Creator*** window. You can easily plug a balloon to an object(see images below).



6. TIPS

- ✓ Always place ***PhysicsStabilizer*** in your scene. This component gives better performances when scene start (check demo scenes to find examples).
- ✓ You need ***KiltUICamera*** attached to one camera of your scene to enable ***RopeCutter.***
- ✓ Check Demo Scenes to understand how to create your scene.
- ✓ You can create blocks prefabs in **KILT->Create->Block**, these blocks can be used to understand mechanics of a complex physics game.
- ✓ This asset was used to implement our game ***BAD BLOCKS***, check it to better understand physics implementations.
- ✓ Use ***SceneManager*** to change scene instead of Application.loadlevel. ***SceneManager*** have better functions/callbacks and can apply an amazing fade in/out effect when changing scene. ***SceneManager*** contains the name of all scenes, so you can retrieve this information in-game.